BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of: Chupa et al.	:	Before the Examiner: Lee, Marina
Serial No.: 10/798,936	:	Group Art Unit: 2192
Filing Date: March 11, 2004	:	
Title: METHOD FOR THE INCREMENTAL DEPLOYMENT OF ENTERPRISE JAVA BEANS	:	IBM Corporation Intellectual Property Law 11400 Burnet Road Austin, Texas 78758

APPEAL BRIEF

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

I. REAL PARTY IN INTEREST

The real party in interest is International Business Machines Corporation, which is the assignee of the entire right, title and interest in the above-identified patent application.

II. RELATED APPEALS AND INTERFERENCES

There are no other appeals or interferences known to Appellants, Appellants' legal representative or assignee which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

III. STATUS OF CLAIMS

Claims 1-7 are pending in the Application. Claims 8-20 were cancelled. Claims 1-7 stand rejected. Claims 1-7 are appealed.

IV. STATUS OF AMENDMENTS

Appellants submitted an amendment (7/17/2008) following receipt of the final office action (4/17/2008) amending the title of the Specification.

V. SUMMARY OF CLAIMED SUBJECT MATTER

Independent Claim 1:

In one embodiment of the present invention, a method for selectively deploying enterprise software comprising for each deployable software component in an preselected input archive file, comparing interfaces for the deployable software component identified in a first descriptor file in the input archive file and a second descriptor file in a preselected output archive file. Specification, page 8, lines 15-26; Figure 2, step 212. The method further comprises if the comparing step miscompares for a first deployable software component, tagging the first deployable software component. Specification, page 8, line 24 – page 9, line 2; Figure 2, step 214. Furthermore, the method comprises if the comparing step miscompares for a second deployable software component, tagging the second deployable software component. Specification, page 8, line 24 – page 9, line 2; Figure 2, step 214. Additionally, the method comprises deploying each tagged deployable software component. Specification, page 9, lines 14-15; Figure 2, step 224.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A. Claims 1, 2, 4 and 7 stand rejected under 35 U.S.C. §102(e) as being anticipated by Garms et al. (U.S. Patent Application Publication No. 2004/0230942) (hereinafter "Garms").

B. Claim 3 stands rejected under 35 U.S.C. §103(a) as being unpatentable over Garms in view of Lagergren (U.S. Patent No. 6,964,042).

C. Claims 5 and 6 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Garms in view of Kovacs et al. (U.S. Patent Application Publication No. 2004/0158571) (hereinafter "Kovacs").

VII. ARGUMENT

A. Claims 1, 2, 4 and 7 are not properly rejected under 35 U.S.C. §102(e) as being anticipated by Garms.

The Examiner has rejected claims 1, 2, 4 and 7 under 35 U.S.C. §102(e) as being anticipated by Garms. Appellants respectfully traverse these rejections for at least the reasons stated below.

For a claim to be anticipated under 35 U.S.C. §102, each and every claim limitation must be found within the cited prior art reference and arranged as required by the claim. M.P.E.P. §2131.

1. Claim 1 is not anticipated by Garms.

Appellants respectfully assert that Garms does not disclose "for each deployable software component in an preselected input archive file, comparing interfaces for the deployable software component identified in a first descriptor file in said input archive file and a second descriptor file in a preselected output archive file" as recited in claim 1. The Examiner cites paragraphs [0020-0021] of Garms as disclosing the above-cited claim limitations. Office Action (10/30/2007), pages 4-5. Appellants respectfully traverse.

Garms instead discloses the step of constructing a deployable modules list in an application. [0020]. Garms further discloses that for J2EE applications, this includes web applications and Enterprise JavaBeans, where the deployable module list can contain at least one deployable module. [0020]. Additionally, Garms discloses the step of examining at least one deployment descriptor for the application to determine if the at least one deployment descriptor contains an entry for each of the at least one deployable module. [0021].

Hence, Garms discloses constructing a deployable modules list in an application as well as examining at least one deployment descriptor for the application to determine if the at least one deployment descriptor contains an entry for each of the at least one deployable module.

There is no language in the cited passages that discloses comparing interfaces. Neither is there any language in the cited passages that discloses comparing interfaces for the deployable software component identified in a first descriptor file in the input archive file and a second descriptor file in a preselected output archive file. Neither is there any language in the cited passages that discloses comparing interfaces for the deployable software component identified in a first descriptor file in the input archive file and a second descriptor file in a preselected output archive file for each deployable software component in a preselected input archive file. Thus, Garms does not disclose all of the limitations of claim 1, and thus Garms does not anticipate claim 1. M.P.E.P. §2131.

In response to Appellants' above arguments, the Examiner cites paragraphs [0019-0021] and [0050] of Garms as disclosing "for each deployable software component in an preselected input archive file, comparing interfaces for the deployable software component identified in a first descriptor file in said input archive file and a second descriptor file in a preselected output archive file" as recited in claim 1. Office Action (4/17/2008), page 6. Appellants respectfully traverse.

Garms instead discloses constructing a deployable modules list in an application as well as examining at least one deployment descriptor for the application to determine if the at least one deployment descriptor contains an entry for each of the at least one deployable module. [0019-0021].

There is no language in the cited passages that discloses comparing interfaces. Neither is there any language in the cited passages that discloses comparing interfaces for the deployable software component identified in a first descriptor file in the input archive file and a second descriptor file in a preselected output archive file. Neither is

there any language in the cited passages that discloses comparing interfaces for the deployable software component identified in a first descriptor file in the input archive file and a second descriptor file in a preselected output archive file for each deployable software component in a preselected input archive file. Thus, Garms does not disclose all of the limitations of claim 1, and thus Garms does not anticipate claim 1. M.P.E.P. §2131.

Further, in response to Appellants' above arguments, the Examiner interprets Garms' teaching of examining at least one deployment descriptor for the application to determine if the at least one deployment descriptor contains an entry for each of the at least one deployable module" as disclosing "comparing interfaces for the deployable software component identified in a first descriptor file in said input archive file and a second descriptor file in a preselected output archive file." Office Action (4/17/2008), page 3. Appellants respectfully traverse.

Determining if the least one deployment descriptor contains an entry for each of the at least one deployable module is not the same as comparing interfaces. Further, the Examiner appears to construe a deployment descriptor as disclosing the claimed input and output archive file. Office Action (4/17/2008), page 3. Claim 1 recites comparing interfaces for the deployable software component identified in a first descriptor file in the input archive file and a second descriptor file in a preselected output archive file. Using the Examiner's interpretation of a deployment descriptor, the Examiner has to provide the teaching of a first descriptor file in the deployment descriptor and a second descriptor file in a separate deployment descriptor. The Examiner has failed to provide such teaching. Hence, Garms does not disclose all of the limitations of claim 1, and thus Garms does not anticipate claim 1. M.P.E.P. §2131.

Additionally, the Examiner appears to interpret the deployable module list of Garms as disclosing the claimed input archive file. Office Action (4/17/2008), page 3. Claim 1 recites comparing interfaces for the deployable software component

identified in a first descriptor file in the input archive file. Using the Examiner interpretation of a deployable module list, the Examiner has to provide the teaching of a first descriptor file in the deployable module list. The Examiner has failed to provide such teaching. Hence, Garms does not disclose all of the limitations of claim 1, and thus Garms does not anticipate claim 1. M.P.E.P. §2131.

Additionally, in connection with the rejection of the above-cited claim limitation, the Examiner equates the deployable module, as taught in Garms, as disclosing the claimed interface. Office Action (4/17/2008), page 4. The Examiner had previously equated the deployable module list of Garms as teaching the claimed preselected input archive file. Garms discloses that the deployable module list contains a deployable module. However, an interface is not contained in a file or a list. Using the Examiner's interpretation of an interface, an interface may be interpreted as an abstract class. An abstract class is a class created as a master structure. See http://www.techweb.com/encyclopedia/defineterm.jhtml?term=abstract+class. Hence, Garms does not disclose all of the limitations of claim 1, and thus Garms does not anticipate claim 1. M.P.E.P. §2131.

Appellants further assert that Garms does not disclose "if the comparing step miscompares for a first deployable software component, tagging said first deployable software component" as recited in claim 1. The Examiner cites paragraphs [0022 and 0023] of Garms as disclosing the above-cited claim limitation. Office Action (10/30/2007), page 5. Appellants respectfully traverse.

Garms instead discloses that if deployable modules are identified without corresponding entries in the application's deployment descriptors, then construct a list of modules to deploy containing each deployable module that was not found in the deployment descriptors. [0022-0023].

There is no language in the cited passages that discloses tagging a first deployable software component if the comparing step miscompares for the first

deployable software component. There is no language in Garms that discloses the comparing step, namely, comparing between the interfaces for the deployable software component in the input archive file and in the output archive file. Instead, Garms discloses determining if deployable modules are identified without corresponding entries in the application's deployment descriptors. Thus, Garms does not disclose all of the limitations of claim 1, and thus Garms does not anticipate claim 1. M.P.E.P. §2131.

In response to Appellants' above arguments, the Examiner cites paragraphs [0021-0024] and [0050] of Garms as disclosing "if the comparing step miscompares for a first deployable software component, tagging said first deployable software component" as recited in claim 1. Office Action (4/17/2008), page 6. Appellants respectfully traverse.

Garms instead discloses that if deployable modules are identified without corresponding entries in the application's deployment descriptors, then construct a list of modules to deploy containing each deployable module that was not found in the deployment descriptors. [0022-0023].

There is no language in the cited passages that discloses tagging a first deployable software component if the comparing step miscompares for the first deployable software component. There is no language in Garms that discloses the comparing step, namely, comparing between the interfaces for the deployable software component in the input archive file and in the output archive file. Instead, Garms discloses determining if deployable modules are identified without corresponding entries in the application's deployment descriptors. Thus, Garms does not disclose all of the limitations of claim 1, and thus Garms does not anticipate claim 1. M.P.E.P. §2131.

Further, in connection with the rejection of the above-cited claim limitation, the Examiner makes the assertion that there is a comparison between module tag to module tag. Office Action (4/17/2008), page 4. The Examiner has not provided any

evidence to support such an assertion. Further, the Examiner has not explained the connection between this statement and the limitation of tagging a first deployable software component if the comparing step miscompares for the first deployable software component. Thus, Garms does not disclose all of the limitations of claim 1, and thus Garms does not anticipate claim 1. M.P.E.P. §2131.

Appellants further assert that Garms does not disclose "if the comparing step miscompares for a second deployable software component, tagging said second deployable software component" as recited in claim 1. The Examiner cites paragraphs [0022 and 0023] of Garms as disclosing the above-cited claim limitation. Office Action (10/30/2007), page 5. Appellants respectfully traverse.

As stated above, Garms instead discloses that if deployable modules are identified without corresponding entries in the application's deployment descriptors, then construct a list of modules to deploy containing each deployable module that was not found in the deployment descriptors. [0022-0023].

There is no language in the cited passages that discloses tagging a second deployable software component if the comparing step miscompares for the second deployable software component. There is no language in Garms that discloses the comparing step, namely, comparing between the interfaces for the deployable software component in the input archive file and in the output archive file. Instead, Garms discloses determining if deployable modules are identified without corresponding entries in the application's deployment descriptors. Thus, Garms does not disclose all of the limitations of claim 1, and thus Garms does not anticipate claim 1. M.P.E.P. §2131.

In response to Appellants' above arguments, the Examiner cites paragraphs [0021-0024] and [0050] of Garms as disclosing "if the comparing step miscompares for a second deployable software component, tagging said second deployable software component" as recited in claim 1. Office Action (4/17/2008), pages 6-7. Appellants respectfully traverse.

Garms instead discloses that if deployable modules are identified without corresponding entries in the application's deployment descriptors, then construct a list of modules to deploy containing each deployable module that was not found in the deployment descriptors. [0022-0023].

There is no language in the cited passages that discloses tagging a second deployable software component if the comparing step miscompares for the second deployable software component. There is no language in Garms that discloses the comparing step, namely, comparing between the interfaces for the deployable software component in the input archive file and in the output archive file. Instead, Garms discloses determining if deployable modules are identified without corresponding entries in the application's deployment descriptors. Thus, Garms does not disclose all of the limitations of claim 1, and thus Garms does not anticipate claim 1. M.P.E.P. §2131.

2. Claims 2, 4 and 7 are not anticipated by Garms for at least the above-stated reasons that claim 1 is not anticipated by Garms.

Claims 2, 4 and 7 each recite combinations of features of independent claim 1, and hence claims 2, 4 and 7 are not anticipated by Garms for at least the above-stated reasons that claim 1 is not anticipated by Garms.

3. Claim 2 is not anticipated by Garms.

Appellants respectfully assert that Garms does not disclose "wherein tagging a deployable software component comprises storing a name of the deployable software component in a file" as recited in claim 2. The Examiner cites paragraphs [0050-0053] of Garms as disclosing the above-cited claim limitation. Office Action (10/30/2007), page 5; Office Action (4/17/2008), page 7. Appellants respectfully traverse.

Garms instead shows that exemplary deployment descriptors might be automatically created and updated by the iterative deployment process. [0051].

Garms further discloses that these examples are taken from a J2EE environment and include application.xml and weblogic-application.xml. [0051].

There is no language in the cited passages that discloses that tagging a deployable software component comprises storing a name of the deployable software component in a file. Thus, Garms does not disclose all of the limitations of claim 2, and thus Garms does not anticipate claim 2. M.P.E.P. §2131.

4. Claim 4 is not anticipated by Garms.

Appellants respectfully assert that Garms does not disclose "if the first descriptor file and second descriptor file compare for the first deployable software component, introspecting a binary class file for the first deployable software component in the input and output archive files; and if, in response to the introspection, a signature or return type of an interface of said binary class files miscompare, tagging the first deployable software component" as recited in claim 4. The Examiner cites paragraphs [0031-0034] of Garms as disclosing the above-cited claim limitations. Office Action (10/30/2007), pages 5-6; Office Action (4/17/2008), page 7. Appellants respectfully traverse.

Garms instead discloses that for each module in the list of deployable modules, the following occurs: construct a list of relevant attributes for the current module; add any of the attributes identified in the previous step that are not included in the corresponding deployment descriptor entry for the current module to the deployment descriptor entry; and change the value of the deployment descriptor entry to match that of the associated deployable module's value if any of the attributes identified in the first step are included in the corresponding deployment descriptor entry for the current module but have a different value. [0031-0034].

Hence, Garms discloses adding any of the attributes that are not included in the corresponding deployment descriptor entry for the current module to the deployment descriptor entry, and changing the value of the deployment descriptor entry to match that of the associated deployable module's value if any of the attributes

are included in the corresponding deployment descriptor entry for the current module but have a different value.

There is no language in the cited passages that discloses introspecting a binary class file for the first deployable software component in the input and output archive files. Neither is there any language in the cited passages that discloses introspecting a binary class file for the first deployable software component in the input and output archive files if the first descriptor file and second descriptor file compare for the first deployable software component. Neither is there any language in the cited passages that discloses tagging the first deployable software component if, in response to the introspection, a signature or return type of an interface of the binary class files miscompare. Thus, Garms does not disclose all of the limitations of claim 4, and thus Garms does not anticipate claim 4. M.P.E.P. §2131.

5. Claim 7 is not anticipated by Garms.

Appellants respectfully assert that Garms does not disclose "wherein the comparing, tagging and deploying steps are performed in response to an execution of a build script invoking a selective deployer utility" as recited in claim 7. The Examiner cites paragraphs [0006-0007] of Garms as disclosing the above-cited claim limitation. Office Action (10/30/2007), page 6; Office Action (4/17/2008), page 8. Appellants respectfully traverse.

Garms instead discloses that developing these deployment descriptors manually is tedious and time consuming. [0006]. Garms further discloses that each time the deployment configuration changes, the deployment descriptors must change to match. [0006]. Additionally, Garms discloses that using incremental application deployment, all the files under development can be modified directly in place on the server's disk. [0007]. Furthermore, Garms discloses that application and module configuration information can be collected from the user as they build their application, e.g. using an Integrated Development Environment (IDE). [0007].

Hence, Garms discloses that each time the deployment configuration changes, the deployment descriptors must change to match.

There is no language in the cited passages that discloses that the comparing, tagging and deploying steps are performed in response to an execution of a build script. Neither is there any language in the cited passages that discloses that the comparing, tagging and deploying steps are performed in response to an execution of a build script invoking a selective deployer utility. Thus, Garms does not disclose all of the limitations of claim 7, and thus Garms does not anticipate claim 7. M.P.E.P. §2131.

B. Claim 3 is not properly rejected under 35 U.S.C. §103(a) as being unpatentable over Garms in view of Lagergren.

1. Garms and Lagergren, taken singly or in combination, do not teach at least the following claim limitations.

Appellants respectfully assert that Garms and Lagergren, taken singly or in combination, do not teach "if the first descriptor file and second descriptor file compare for the first deployable software component, comparing a size of a binary class file for the first deployable software component in the input and output archive files; and if the size of said binary class files miscompare, tagging the first deployable software component" as recited in claim 3. The Examiner cites steps 30-31 of Figure 4; column 5, lines 26-67 and column 6, lines 1-9 of Lagergren as teaching the above-cited claim limitations. Office Action (10/30/2007), page 7; Office Action (4/17/2008), page 9. Appellants respectfully traverse.

Lagergren instead teaches how a dynamic size metric is calculated utilizing both predetermined and system-determined factors. Column 5, lines 27-29. Lagergren further teaches that the process begins with a request to calculate a dynamic size metric for the current application code. Column 5, lines 30-32. In addition, Lagergren teaches that the system utilizes a set of factors in order to calculate the dynamic size metric. Column 5, lines 32-33. Furthermore, Lagergren

teaches that the static size metric is replaced with a dynamic size metric, which is a function of factors that the feedback mechanism supplies after an analysis of a specific size of code has been performed. Column 5, lines 48-52. Further, Lagergren teaches determining intermediate size metrics which are adjusted when necessary to account for architecture specific requirements. Column 6, lines 2-5.

Hence, Lagergren teaches the static size metric is replaced with a dynamic size metric, which is a function of factors that the feedback mechanism supplies after an analysis of a specific size of code has been performed.

There is no language in the cited passages that teaches comparing a size of a binary class file for the first deployable software component in the input and output archive files. Neither is there any language in the cited passages that teaches comparing a size of a binary class file for the first deployable software component in the input and output archive files if the first descriptor file and second descriptor file compare for the first deployable software component. Neither is there any language in the cited passages that teaches tagging the first deployable software component. Neither is there any language in the cited passages that teaches tagging the first deployable software component if the size of the binary class files miscompare.

Therefore, the Examiner has not presented a *prima facie* case of obviousness in rejecting claim 3, since the Examiner is relying upon incorrect, factual predicates in support of the rejection. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1455 (Fed. Cir. 1998).

2. Examiner's reasoning for modifying Garms with Lagergren to include the missing claim limitations of claim 3 is insufficient to establish a *prima facie* case of obviousness.

Most if not all inventions arise from a combination of old elements. *See In re Rouffet*, 47 U.S.P.Q.2d 1453, 1457 (Fed. Cir. 1998). Obviousness is determined from the vantage point of a hypothetical person having ordinary skill in the art to which the patent pertains. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1457 (Fed. Cir. 1998). Therefore, an Examiner may often find every element of a claimed invention in the prior art. *Id.*

However, identification in the prior art of each individual part claimed is insufficient to defeat patentability of the whole claimed invention. *See Id.* In order to establish a *prima facie* case of obviousness, the Examiner must show reasons that the skilled artisan, confronted with the same problems as the inventor and with no knowledge of the claimed invention, would select the elements from the cited prior art references for combination in the manner claimed. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1458 (Fed. Cir. 1998). The Examiner must provide articulated reasoning with some rational underpinning to support the legal conclusion of obviousness. *In re Kahn*, 441 F.3d 977, 988 (Fed. Cir. 2006) (cited approvingly in *KSR International Co. v. Teleflex Inc.*, 82 U.S.P.Q.2d 1385, 1396 (U.S. 2007)).

As understood by Appellants, the Examiner admits that Garms does not teach "if the first descriptor file and second descriptor file compare for the first deployable software component, comparing a size of a binary class file for the first deployable software component in the input and output archive files; and if the size of said binary class files miscompare, tagging the first deployable software component" as recited in claim 3. Office Action (10/30/2007), pages 6-7; Office Action (4/17/2008), page 8. The Examiner asserts that Lagergren teaches the above-cited claim limitations. Office Action (10/30/2007), page 7; Office Action (4/17/2008), page 9. The Examiner's reasoning for modifying Garms with Lagergren to include the above-cited claim limitations is to "optimize performance of particular module environment (e.g., EJB, Web). – See (Lagergren, col. 2: 10-20)." *Id.* The Examiner's reasoning is insufficient to establish a *prima facie* case of obviousness in rejecting claim 3.

As stated above, the Examiner cites column 2, lines 10-20 of Lagergren as support for the Examiner's reasoning for modifying Garms with Lagergren to include the above-cited missing claim limitations of claim 3. Lagergren teaches that traditional systems rarely take into account any form of dynamic optimization, such as how an application code may be optimized to better address the running environment. Column 2, lines 10-13. Lagergren further teaches that the traditional method of code optimization is to monitor the size of the application code in memory

and to perform optimizations to keep the application code within a certain memory size range. Column 2, lines 13-16. Additionally, Lagergren teaches that none of these techniques of code optimization address the individual needs of particular environments, or of virtual machines running in these environments. Column 2, lines 16-19. Hence, Lagergren teaches that prior art techniques of code optimization did not address the individual needs of particular environments, or of virtual machines running in these environments.

There is no language in Lagergren (and in particular column 2, lines 10-20) that makes any suggestion to: (1) compare a size of a binary class file for the first deployable software component in the input and output archive files if the first descriptor file and second descriptor file compare for the first deployable software component; and (2) tag the first deployable software component if the size of the binary class files miscompare (missing claim limitations) in order to optimize the performance of a particular module environment (Examiner's reasoning). The Examiner has to provide some rational connection between the cited passage that is the source of the Examiner's reasoning and the above-cited missing claim limitations. The Examiner has cited to a passage in Lagergren that teaches the deficiencies in the prior art. The Examiner's source of reasoning (column 2, lines 10-20 of Lagergren) does not provide reasons as to why one skilled in the art would modify Garms to include the above-cited missing claim limitations of claim 3. Accordingly, the Examiner has not presented a *prima facie* case of obviousness for rejecting claim 3. *KSR International Co. v. Teleflex Inc.*, 82 U.S.P.Q.2d 1385, 1396 (U.S. 2007).

Further, the Examiner's reasoning ("optimize performance of particular module environment") does not provide reasons, as discussed further below, that the skilled artisan, confronted with the same problems as the inventor and with no knowledge of the claimed invention, would modify Garms to include the above-indicated missing claim limitations of claim 3. Accordingly, the Examiner has not presented a *prima facie* case of obviousness for rejecting claim 3. *KSR International*

Co. v. Teleflex Inc., 82 U.S.P.Q.2d 1385, 1396 (U.S. 2007); *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1458 (Fed. Cir. 1998).

Garms addresses the problem of long delays in the development cycle in developing software. [0004-0006]. The Examiner has not provided any reasons as to why one skilled in the art would modify Garms (which addresses the problem of long delays in the development cycle in developing software) to: (1) compare a size of a binary class file for the first deployable software component in the input and output archive files if the first descriptor file and second descriptor file compare for the first deployable software component; and (2) tag the first deployable software component if the size of the binary class files miscompare (missing claim limitations). The Examiner's rationale ("optimize performance of particular module environment") does not provide such reasoning.

Why would the reason to modify Garms (whose purpose is to address the problem of long delays in the development cycle in developing software) to: (1) compare a size of a binary class file for the first deployable software component in the input and output archive files if the first descriptor file and second descriptor file compare for the first deployable software component; and (2) tag the first deployable software component if the size of the binary class files miscompare (missing claim limitations) be to optimize the performance of the particular module environment?

There are many ways to optimize the performance of the particular module environment. The Examiner though must explain as to why in particular one skilled in the art would modify Garms to: (1) compare a size of a binary class file for the first deployable software component in the input and output archive files if the first descriptor file and second descriptor file compare for the first deployable software component; and (2) tag the first deployable software component if the size of the binary class files miscompare (missing claim limitations) in order to optimize the performance of the particular module environment.

Hence, the Examiner's rationale does not provide reasons that the skilled artisan, confronted with the same problems as the inventor and with no knowledge of the claimed invention, would modify Garms to include the above-cited missing claim limitations of claim 3. Accordingly, the Examiner has not presented a *prima facie* case of obviousness for rejecting claim 3. *KSR International Co. v. Teleflex Inc.*, 82 U.S.P.Q.2d 1385, 1396 (U.S. 2007); *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1458 (Fed. Cir. 1998).

C. Claims 5 and 6 are not properly rejected under 35 U.S.C. §103(a) as being unpatentable over Garms in view of Kovacs.

1. Garms and Kovacs, taken singly or in combination, do not teach at least the following claim limitations.

a. Claim 5 is patentable over Garms in view of Kovacs.

Appellants respectfully assert that Garms and Kovacs, taken singly or in combination, do not teach "opening said preselected output archive file; and if the step of opening the preselected output archive fails, tagging each deployable software component in the input archive file" as recited in claim 5. The Examiner cites element 302 in Figure 3 and paragraphs [0020-0021] of Kovacs as teaching the above-cited claim limitations. Office Action (10/30/2007), page 8; Office Action (4/17/2008), page 9. Appellants respectfully traverse.

Kovacs instead teaches that the builder invokes validator 302 to locate errors within deployment descriptor files (e.g., incorrect CMP field name, etc.). [0020]. Kovacs further teaches that when compiler 304 determines that there is an error in an deployment descriptor file, it can create an Error object to store an error code and/or message, the identifier of a node and corresponding field in the resource hierarchy 102 to which the error pertains, and/or an XML type of the field. [0020]. Kovacs further teaches that validator 302 and/or compiler 304 can display human-readable error messages corresponding to each error object in message area 106. [0021].

Hence, Kovacs teaches that when the compiler determines that there is an error in an deployment descriptor file, it can create an error object to store an error code and/or message.

There is no language in the cited passages that teaches opening the preselected output archive file. Neither is there any language in the cited passages that teaches tagging each deployable software component in the input archive file. Neither is there any language in the cited passages that teaches tagging each deployable software component in the input archive file if the step of opening the preselected output archive fails.

Therefore, the Examiner has not presented a *prima facie* case of obviousness in rejecting claim 5, since the Examiner is relying upon incorrect, factual predicates in support of the rejection. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1455 (Fed. Cir. 1998).

b. Claim 6 is patentable over Garms in view of Kovacs.

Appellants respectfully assert that Garms and Kovacs, taken singly or in combination, do not teach "wherein the step of tagging each deployable software component is performed in response to the step of opening the preselected output archive throwing an exception" as recited in claim 6. The Examiner cites paragraphs [0020-0021] of Kovacs as teaching the above-cited claim limitations. Office Action (10/30/2007), page 8; Office Action (4/17/2008), page 10. Appellants respectfully traverse.

As stated above, Kovacs instead teaches that the builder invokes validator 302 to locate errors within deployment descriptor files (e.g., incorrect CMP field name, etc.). [0020]. Kovacs further teaches that when compiler 304 determines that there is an error in an deployment descriptor file, it can create an Error object to store an error code and/or message, the identifier of a node and corresponding field in the resource hierarchy 102 to which the error pertains, and/or an XML type of the field. [0020]. Kovacs further teaches that validator 302 and/or compiler 304 can display human-

readable error messages corresponding to each error object in message area 106. [0021].

There is no language in the cited passages that teaches that the step of tagging each deployable software component is performed in response to the step of opening the preselected output archive throwing an exception. Therefore, the Examiner has not presented a *prima facie* case of obviousness in rejecting claim 6, since the Examiner is relying upon incorrect, factual predicates in support of the rejection. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1455 (Fed. Cir. 1998).

2. Examiner's reasoning for modifying Garms with Kovacs to include the missing claim limitations of claim 5 is insufficient to establish a *prima facie* case of obviousness.

As stated above, most if not all inventions arise from a combination of old elements. *See In re Rouffet*, 47 U.S.P.Q.2d 1453, 1457 (Fed. Cir. 1998). Obviousness is determined from the vantage point of a hypothetical person having ordinary skill in the art to which the patent pertains. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1457 (Fed. Cir. 1998). Therefore, an Examiner may often find every element of a claimed invention in the prior art. *Id.* However, identification in the prior art of each individual part claimed is insufficient to defeat patentability of the whole claimed invention. *See Id.* In order to establish a *prima facie* case of obviousness, the Examiner must show reasons that the skilled artisan, confronted with the same problems as the inventor and with no knowledge of the claimed invention, would select the elements from the cited prior art references for combination in the manner claimed. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1458 (Fed. Cir. 1998). The Examiner must provide articulated reasoning with some rational underpinning to support the legal conclusion of obviousness. *In re Kahn*, 441 F.3d 977, 988 (Fed. Cir. 2006) (cited approvingly in *KSR International Co. v. Teleflex Inc.*, 82 U.S.P.Q.2d 1385, 1396 (U.S. 2007)).

As understood by Appellants, the Examiner admits that Garms does not teach "opening said preselected output archive file; and if the step of opening the preselected output archive fails, tagging each deployable software component in the input archive file" as recited in claim 5. Office Action (10/30/2007), pages 7-8; Office Action (4/17/2008), page 9. The Examiner asserts that Kovacs teaches the above-cited claim limitations. Office Action (10/30/2007), page 8; Office Action (4/17/2008), page 9. The Examiner's reasoning for modifying Garms with Kovacs to include the above-cited claim limitations is "to identify the input errors and offer the suggesting solution to those errors (see Kovacs, page 2, [0021])." Office Action (10/30/2007), page 8; Office Action (4/17/2008), page 10. The Examiner's reasoning is insufficient to establish a *prima facie* case of obviousness in rejecting claim 5.

As stated above, the Examiner cites paragraph [0021] of Kovacs as support for the Examiner's reasoning for modifying Garms with Kovacs to include the above-cited missing claim limitations of claim 5. Kovacs teaches that validator 302 and/or compiler 304 can display human-readable error messages corresponding to each error object in message area 106. [0021]. Kovacs further teaches that validator 302 can offer suggestions to the user for correcting the field value via a pop-up window or some other notification means. [0021]. Hence, Kovacs teaches displaying human-readable error messages corresponding to each error object in a message area as well as offering suggestions for correcting the field value via a pop-up window or some other notification means.

There is no language in Kovacs (and in particular paragraph [0021]) that makes any suggestion to: (1) open the preselected output archive file; and (2) tag each deployable software component in the input archive file if the step of opening the preselected output archive fails (missing claim limitations) in order to identify the input errors and offer the suggesting solution to those errors (Examiner's reasoning). The Examiner has to provide some rational connection between the cited passage that is the source of the Examiner's reasoning and the above-cited missing claim limitations. The Examiner's source of reasoning (paragraph [0021] of Kovacs) does

not provide reasons as to why one skilled in the art would modify Garms to include the above-cited missing claim limitations of claim 5. Accordingly, the Examiner has not presented a *prima facie* case of obviousness for rejecting claim 5. *KSR International Co. v. Teleflex Inc.*, 82 U.S.P.Q.2d 1385, 1396 (U.S. 2007).

Further, the Examiner's reasoning ("to identify the input errors and offer the suggesting solution to those errors") does not provide reasons, as discussed further below, that the skilled artisan, confronted with the same problems as the inventor and with no knowledge of the claimed invention, would modify Garms to include the above-indicated missing claim limitations of claim 5. Accordingly, the Examiner has not presented a *prima facie* case of obviousness for rejecting claim 5. *KSR International Co. v. Teleflex Inc.*, 82 U.S.P.Q.2d 1385, 1396 (U.S. 2007); *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1458 (Fed. Cir. 1998).

Garms addresses the problem of long delays in the development cycle in developing software. [0004-0006]. The Examiner has not provided any reasons as to why one skilled in the art would modify Garms (which addresses the problem of long delays in the development cycle in developing software) to: (1) open the preselected output archive file; and (2) tag each deployable software component in the input archive file if the step of opening the preselected output archive fails (missing claim limitations). The Examiner's rationale ("to identify the input errors and offer the suggesting solution to those errors") does not provide such reasoning.

Why would the reason to modify Garms (whose purpose is to address the problem of long delays in the development cycle in developing software) to: (1) open the preselected output archive file; and (2) tag each deployable software component in the input archive file if the step of opening the preselected output archive fails (missing claim limitations) be to identify the input errors and offer the suggesting solution to those errors?

Garms is not concerned with identifying the input errors and offering the suggesting solution to those errors. The Examiner cannot completely ignore the

teachings of Garms in concluding it would have been obvious to modify Garms to include the above-cited missing claim limitations of claim 5. Further, what is the rational connection between opening the preselected output archive file (missing claim limitation) and identifying the input errors and offering the suggesting solution to those errors (Examiner's reasoning)? Further, what is the rational connection between tagging each deployable software component in the input archive file if the step of opening the preselected output archive fails (missing claim limitation) and identifying the input errors and offering the suggesting solution to those errors (Examiner's reasoning)?

Hence, the Examiner's rationale does not provide reasons that the skilled artisan, confronted with the same problems as the inventor and with no knowledge of the claimed invention, would modify Garms to include the above-cited missing claim limitations of claim 5. Accordingly, the Examiner has not presented a *prima facie* case of obviousness for rejecting claim 5. *KSR International Co. v. Teleflex Inc.*, 82 U.S.P.Q.2d 1385, 1396 (U.S. 2007); *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1458 (Fed. Cir. 1998).

3. Examiner fails to provide a rational underpinning for modifying Garms with Kovacs to include the missing claim limitation of claim 6.

As stated above, most if not all inventions arise from a combination of old elements. *See In re Rouffet*, 47 U.S.P.Q.2d 1453, 1457 (Fed. Cir. 1998). Obviousness is determined from the vantage point of a hypothetical person having ordinary skill in the art to which the patent pertains. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1457 (Fed. Cir. 1998). Therefore, an Examiner may often find every element of a claimed invention in the prior art. *Id.* However, identification in the prior art of each individual part claimed is insufficient to defeat patentability of the whole claimed invention. *See Id.* In order to establish a *prima facie* case of obviousness, the Examiner must show reasons that the skilled artisan, confronted with the same problems as the inventor and with no knowledge of the claimed invention, would

select the elements from the cited prior art references for combination in the manner claimed. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1458 (Fed. Cir. 1998). The Examiner must provide articulated reasoning with some rational underpinning to support the legal conclusion of obviousness. *In re Kahn*, 441 F.3d 977, 988 (Fed. Cir. 2006) (cited approvingly in *KSR International Co. v. Teleflex Inc.*, 82 U.S.P.Q.2d 1385, 1396 (U.S. 2007)).

As understood by Appellants, the Examiner admits that Garms does not teach the claim limitation of claim 6. Office Action (10/30/2007), page 8; Office Action (4/17/2008), page 10. The Examiner asserts that Kovacs teaches the limitation of claim 6. *Id.* However, the Examiner has not provided any rational underpinning for modifying Garms with Kovacs to include the above-cited missing claim limitation. Hence, the Examiner has not provided a *prima facie* case of obviousness in rejecting claim 6. *KSR International Co. v. Teleflex Inc.*, 82 U.S.P.Q.2d 1385, 1396 (U.S. 2007); M.P.E.P. §2143.

VIII. CONCLUSION

For the reasons noted above, the rejections of claims 1-7 are in error. Appellants respectfully request reversal of the rejections and allowance of claims 1-7.

Respectfully submitted,

WINSTEAD P.C.

Attorneys for Appellants

By: _____

Robert A. Voigt, Jr.
Reg. No. 47,159

P.O. Box 50784
Dallas, Texas 75201
(512) 370-2832

CLAIMS APPENDIX

1. A method for selectively deploying enterprise software comprising:

for each deployable software component in an preselected input archive file, comparing interfaces for the deployable software component identified in a first descriptor file in said input archive file and a second descriptor file in a preselected output archive file;

if the comparing step miscompares for a first deployable software component, tagging said first deployable software component;

if the comparing step miscompares for a second deployable software component, tagging said second deployable software component; and

deploying each tagged deployable software component.

2. The method of claim 1 wherein tagging a deployable software component comprises storing a name of the deployable software component in a file.

3. The method of claim 1 further comprising:

if the first descriptor file and second descriptor file compare for the first deployable software component, comparing a size of a binary class file for the first deployable software component in the input and output archive files; and

if the size of said binary class files miscompare, tagging the first deployable software component.

4. The method of claim 1 further comprising:

if the first descriptor file and second descriptor file compare for the first deployable software component, introspecting a binary class file for the first deployable software component in the input and output archive files; and

if, in response to the introspection, a signature or return type of an interface of said binary class files miscompare, tagging the first deployable software component.

5. The method of claim 1 further comprising:

opening said preselected output archive file; and

if the step of opening the preselected output archive fails, tagging each deployable software component in the input archive file.

6. The method of claim 5 wherein the step of tagging each deployable software component is performed in response to the step of opening the preselected output archive throwing an exception.

7. The method of claim 1 wherein the comparing, tagging and deploying steps are performed in response to an execution of a build script invoking a selective deployer utility.

EVIDENCE APPENDIX

No evidence was submitted pursuant to §§1.130, 1.131, or 1.132 of 37 C.F.R. or of any other evidence entered by the Examiner and relied upon by Appellants in the Appeal.

RELATED PROCEEDINGS APPENDIX

There are no related proceedings to the current proceeding.

Austin_1 543228v.1